Save Distance Save Time - Effect of spatial data structures in effective retrieval of traffic data

ANCY PHILIP | Junior, Anna University

Introduction

In this world where “Speed is money”, we come across various situations where we need information to commute .We may need to identify the shortest route ,the current traffic and hence choose the best path. The data-structure for storing this information does play a crucial role in effectively retrieving the information on the server. R * trees can be effectively used for this. After getting the bounding rectangle from the map we can apply the All Pair Shortest Path algorithm to retrieve the shortest route. Using IoV (internet of vehicles) technology we can combine the current traffic information to select the best route.

What is - Iov and APSP

The Internet of Vehicles (IoV) is the confluence of the Internet of Things IoT and Mobile Internet. IoV technology refers to dynamic mobile communication systems which exchange data between public networks and vehicles. We can thus get information on roads, vehicles and their surroundings. In addition it allows for processing , gauging , sharing and provides information for various applications.

The **All pair shortest path algorithm** identifies the shortest path between all pair of vertices in a graph. This is similar to the idea of finding the shortest path between intersections on a road map (the nodes correspond to intersections and the cost of the edges correspond to the road’s length.)

R* trees

What is R* tree ?

- **R*-trees** are a variant of R-trees used for indexing spatial information.
- It uses a combination of a revised node split algorithm and the concept of forced reinsertion at node overflow.

PROS

- Better query performance
- Minimization of overlap.
- Minimization of coverage.
- Improved split heuristic.
- Reinsertion method optimizes the existing tree.
- Efficiently supports point and spatial data at the same time.
- For leaf nodes, overlap is minimized, while for inner nodes, enlargement and area are minimized.

CONS

- They have slightly higher construction cost than standard R-trees, as the data may need to be reinserted;
- The complexity of range search is O(log $_m$ N)

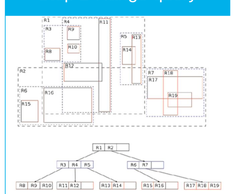
The proposed idea

Step 1-IOV and adjacency matrix

w(u,v)	a	b	c	d	d	f
a	0	4	2	INF	INF	INF
b	4	0	1	5	INF	INF
c	2	1	0	8	10	INF
d	INF	5	8	0	2	6
e	INF	INF	10	2	0	3
f	INF	INF	INF	6	3	0

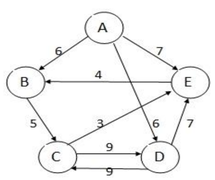
- The current state of traffic continuously obtained from the IoV is mapped onto a graph of the road map .This database could be stored on a cloud infrastructure and efficient parallel implementations could be used for quick mapping of this real time data.
- A matrix is created to contain the nodes , the distance between them and the current speed.

Step 2-Range query



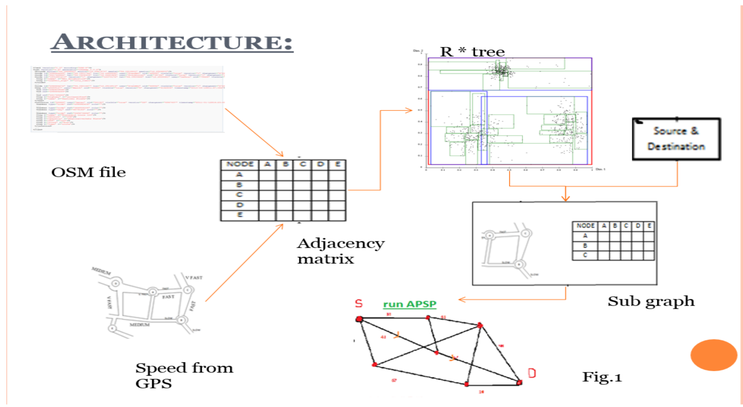
- Based on the users source-destination the R*tree is range queried .
- The bounding rectangle can be identified in O(log$_d$n) .
- From this bounding rectangle a sub-graph is generated.

Step 3-APSP



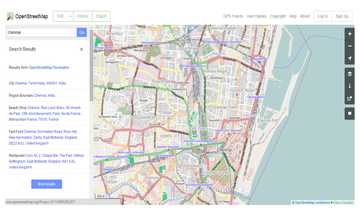
- (APSP)algorithm (Floyd’s) is run on the subset matrix. This will explore shortest path for every pair of vertices.
- The output is finally displayed to the user in response to the query for the the best route between the indicated ‘Source and Destination’.

Architecture



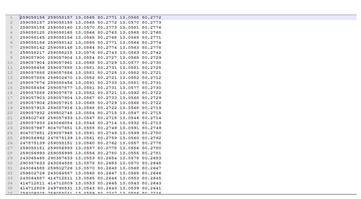
Implementation

- OSM and PARSING
  - Open Street Maps is a collaborative project which provides free open source map data. The data is crowd sourced via hand held GPS, digital camera and voice recorders. The map data is stored in an OSM format which has tags like nodes ,ways ,relations. The Chennai map of the OSM is considered here.
  - Imposm parser is a python library that parses OSM data which is in XML format . Using this library, the nodes are extracted from the XML format of the Chennai OSM.
- A 25000 x 25000 matrix is created to contain the nodes , the distance between them and the current speed.
- The Haversine formula is used to find the distance between the two nodes in a WAY tag in the OSM file on the basis of their corresponding lat-long. The output file of the OSM parse is used to enter the corresponding data in the adjacency matrix.
- As real time traffic data is unavailable for Chennai as a whole, random data is generated to produce speeds between numbers 1-3 depicting slow ,medium ,fast in increasing order. If there no way exists between the nodes, speed of 0 and a distance of infinity is inserted in the matrix.
- And APSP was ran on it.



OPEN STREET MAP-CHENNAI

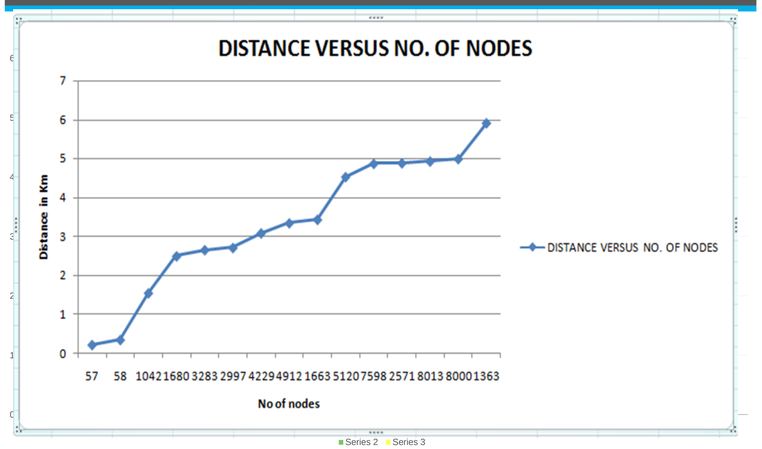
XML FILE OF CHENNAI OSM



RESULT OF PARSING

OUTPUT OF RANGE QUERY

Experimental Results



Here we see that as the Distance between the nodes decreases the no. of nodes in the range query decreases. Thus the APSP is run on this reduced no. of nodes than the entire set of nodes in the original graph giving rise to efficiency in the whole process.

Conclusion

Dealing with traffic is an everyday problem. Finding solutions to this is an absolute challenge for the progress of this ever tech-booming world. This novel idea definitely aids this by supplying the user with shortest routes. Thus the best route can be chosen from the options displayed based on traffic information. This simple idea can be used not only to help the users get over this but also aid the government in traffic and transportation planning based on the data supplied from sensors.

Likewise it can be extrapolated to include the starting time and the system can predict the destination arrival time (in terms of current traffic) and predict the start time to reach the destination at a specified time. Or the system can even output routes which fits inside a duration of time allotted by the user to reach his choice of destination.

Future enhancements

- The most widely used technique for obtaining information about the current traffic is by the employment of GPS technology .Hence the speed factor randomly generated can be replaced by the dynamic GPS data in order to make it a real time application.
- By pulling a sub-graph based on the MBR we might lose track of the otherwise shortest or fastest way existing between the source-destination pair owing to criteria like traffic.
- Adjacency matrix for maps are generally sparse in nature. Implementing adjacency list can reduce the space.
- Implementing parallel algorithms to make data access quicker by the use of CUDA or HADOOP clusters.
- Usage of existing routing libraries with the current data structures to increase efficiency.